DEEP SEARCH IN LINGUISTIC GEOMETRY

DRAFT

Boris Stilman
Department of Computer Science & Engineering, University of Colorado at Denver
Campus Box 109, Denver, CO 80217-3364
Email: bstilman@cse.cudenver.edu

Abstract. This paper is a new step in the development and application of the Linguistic Geometry. We investigate heuristics extracted in the form of hierarchical networks of planning paths of autonomous agents. Employing Linguistic Geometry tools the dynamic hierarchy of networks is represented as a hierarchy of formal attribute languages. The main ideas of this methodology are shown in this paper on the new pilot example of the solution of the extremely complex 2D optimization problem for the autonomous vehicles for the aerospace problem domain. This example demonstrates the dramatic reduction of search in comparison with conventional search algorithms.

Keywords: Artificial Intelligence, Heuristic Search, Linguistic Geometry, Multiagent Systems, Two Player Games.

1. Introduction

Aerospace problems such as long and short-range mission planning, especially for autonomous navigation, scheduling, aerospace robot control, long-range satellite service, aerospace combat operations control, etc. can be formally represented as reasoning about complex large-scale control systems. The field of efficient aerospace control systems needs new technology from the science of artificial intelligence (Rodin, 1988; Lirov, Rodin et al., 1988).

The classic approach based on the theory of Differential Games alone is insufficient, especially in case of dynamic, multiagent models (Garcia-Ortiz et al., 1993). Following (Rodin, 1988; Shinar, 1990) discrete-event modeling of complex control systems can be implemented as a purely interrogative simulation. These techniques can be based on generating geometrically meaningful states rather than time increments with due respect to the timeliness of actions. By discretizing time, a finite game tree can be obtained. The nodes of the tree represent the states of the game, where the players can select their controls for a given period of time. It is also possible that players do not make their decisions simultaneously and in this case, the respective moves of the two sides can be easily distinguished. Thus, the branches of the tree are the moves in the game space. The pruning of such tree is the basic task of heuristic search techniques. Interrogative approach to control problems offers much faster execution and clearer simulator definition (Lirov et al., 1988). For this kind of approach a series of hierarchical dynamic multiagent goal-oriented systems should be developed and investigated.

There are many such problems where human expert skills in reasoning about complex goal-oriented systems are incomparably higher than the level of modern computing systems. Unfortunately, problems of tactics planning and automatic control of autonomous agents such as aerospace vehicles, space stations and robots with cooperative and opposing interests are of the type where human problem-solving skills can not be directly applied. Moreover, there are no highly-skilled human experts in these fields ready to substitute for robots (on a virtual model) or transfer their knowledge to them. There is no grand-master in robot control, although, of course, the knowledge of existing experts in this field should not be neglected – it is even more valuable. Due to the special significance of these problems and the fabulous costs of mistakes, the quality of solutions must be very high and usually subject to continuous improvement.

It is very important to study human expert reasoning about similar complex systems in the areas where the results are successful, in order to discover the keys to success, and then apply and adopt these keys to the new, as yet, unsolved problems, and first and foremost to the aerospace critical complex systems. It should be considered as investigation, development, and consequent expansion of advanced human expert skills into new areas.

2. Background

The difficulties we encounter trying to find the optimal operation for real-world complex control systems are well known. While the formalization of the problem, as a rule, is not difficult, an algorithm that finds its solution usually results in the search of many variations. For small-dimensional "toy" problems a solution can be obtained; however, for most real-world problems the dimension increases and the number of variations increases significantly, usually exponentially, as a function of dimension (Garey and Johnson, 1991). Thus, most real-world search problems are not solvable with the help of exact algorithms in a reasonable amount of time. This becomes increasingly critical for the real-time aerospace autonomous and semiautonomous vehicles and robots (Lirov et al., 1988; Strosnider and Paul, 1994).

There have been many attempts to find the optimal (suboptimal) operation for real-world complex systems, in particular, for aerospace applications (Leitmann, 1990; Drabble, 1991; Pigeon et al., 1992).

One of the basic ideas is to decrease the dimension of the real-world system following the approach of a *human expert in the field*, by breaking the system into smaller subsystems. This process of decomposition can be applied recursively until we end up with a collection of basic subproblems that can be treated (in some sense) independently. These ideas have been implemented for many problems with varying degrees of success (see, e.g., Albus, 1991; Knoblock, 1990; Mesarovich et al, 1989; Botvinnik, 1984). Implementations based on the formal theories of linear and nonlinear planning meet hard efficiency problems (McAllester and Rosenblitt, 1991; Chapman, 1987; Nilsson, 1980; Stefik, 1981; Sacerdoti, 1975). An efficient planner requires an intensive use of heuristic knowledge. Moreover, it is possible to use both dynamic and static heuristic knowledge in reducing the search variations. The dynamic knowledge can be acquired during the run time and immediately applied for search reduction (Strosnider and Paul, 1994).

In the 1960's, a formal syntactic approach to the investigation of properties of natural language resulted in the fast development of a theory of formal languages by Chomsky (1963), Ginsburg (1966), and others. This development provided an interesting opportunity for dissemination of this approach to different areas. In particular, there came an idea of analogous linguistic representation of images. This idea was successfully developed into syntactic methods of pattern recognition by Fu (1982), Narasimhan (1966), and Pavlidis (1977), and picture description languages by Shaw (1969), Feder (1971), and Rosenfeld (1979).

Searching for adequate mathematical tools formalizing human heuristics of dynamic hierarchies, we have transformed the idea of linguistic representation of complex real-world and artificial images into the idea of similar representation of complex hierarchical systems (Stilman, 1985). However, the appropriate languages should possess more sophisticated attributes than languages usually used for pattern description. The origin of such languages can be traced back to the research on programmed attribute grammars by Knuth (1968), Rozenkrantz (1969).

A mathematical environment (a "glue") for the formal implementation of this approach was developed following the theories of formal problem solving and planning by Nilsson (1980), Fikes and Nilsson (1971), Sacerdoti (1975), McCarthy (1980), McCarthy and Hayes (1969), and others based on first order predicate calculus.

In the beginning of 80's Botvinnik, Stilman, and others developed one of the most interesting and powerful heuristic hierarchical models. It was successfully applied to scheduling, planning, control, and computer chess. The hierarchical networks were introduced in (Botvinnik, 1984; Stilman, 1977) in the form of ideas, plausible discussions, and program implementations (see below). We consider this model as an ideal case for transferring the developed search heuristics to the aerospace domain employing formal linguistic tools.

An application of the developed model to a chess domain was implemented in full as program PIONEER (Botvinnik, 1984). Similar heuristic model was implemented for power equipment maintenance in a number of computer programs being used for maintenance scheduling all over the USSR (Botvinnik et al., 1983; Reznitskiy and Stilman, 1983; Stilman, 1985, 1993a). All these earlier developed programs were the direct implementations of the specific dynamic hierarchies of subsystems. The first pilot implementation of the elements of the generic hierarchy of formal languages for the 2D-space case was done at the University of Colorado at Denver by King (1993) and Mathews (1993) employing CLIPS tools (Giarratano, 1991) and C language, respectively.

The results shown by these programs in solving complex chess and scheduling problems indicate that implementations of the dynamic hierarchy resulted in the extremely goal-driven algorithms generating search trees with a branching factor close to 1.

In order to discover the inner properties of human expert heuristics, which have been successful in a certain class of complex control systems, we develop a formal theory, the so-called *Linguistic Geometry* (Stilman, 1993-94). This research includes the development of syntactic tools for *knowledge representation* and *reasoning* about large-scale hierarchical complex systems. It relies on the formalization of *search heuristics*, which allow one to decompose complex system into a hierarchy of subsystems, and thus solve intractable problems by reducing the search. These *hierarchical images* in the form of networks of paths were extracted from the expert vision of the problem. The hierarchy of subsystems is represented as a *hierarchy of formal attribute languages*.

3. Knowledge Representation in Linguistic Geometry

A *Complex System* is the following eight-tuple:

$$< X, P, R_p, \{ON\}, v, S_i, S_t, TR>, \text{ where}$$

$X=\{x_i\}$ is a finite set of *points*;

$P=\{p_i\}$ is a finite set of *elements*; P is a union of two non-intersecting subsets P_1 and P_2;

$R_p(x, y)$ is a set of binary relations of *reachability* in X (x and y are from X, p from P);

ON(p)=x, where ON is a partial function of *placement* from P into X;

v is a function on P with positive integer values describing the *values* of elements.

The Complex System searches the state space, which should have initial and target states;

S_i and S_t are the descriptions of the *initial* and *target* states in the language of the first order predicate calculus, which matches with each relation a certain Well-Formed Formula (WFF). Thus, each state from S_i or S_t is described by a certain set of WFF of the form $\{ON(p_j) = x_k\}$;

TR is a set of operators, TRANSITION(p, x, y), of transitions of the System from one state to another one. These operators describe the transition in terms of two lists of WFF (to be removed from and added to the description of the state), and of WFF of applicability of the transition. Here,

Remove list: ON(p)=x, ON(q)=y;
Add list: ON(p)=y;
Applicability list: $(ON(p)=x)\wedge R_p(x,y)$,

where p belongs to P_1 and q belongs to P_2 or vice versa. The transitions are carried out with participation of one or many elements p from P_1 and P_2.

According to the definition of the set P, the elements of the System are divided into two subsets P_1 and P_2. They might be considered as units moving along the reachable points. Element p can move from point x to point y if these points are reachable, i.e., $R_p(x, y)$ holds. The current location of each element is described by the equation ON(p)=x. Thus, the description of each state of the System $\{ON(p_j)=x_k\}$ is the set of descriptions of the locations of the elements. The operator TRANSITION(p, x, y) describes the change of the state of the System caused by the move of the element p from point x to point y. The element q from point y must be withdrawn (eliminated) if p and q do not belong to the same one of the two subsets P_1 and P_2.

The problem of the optimal operation of the System is considered as a search for the optimal sequence of transitions leading from one of the initial states of S_i to a target state S of S_t.

It is easy to show formally that a robotic system can be considered as a Complex System (see below). Many different technical and human society systems (including military battlefield systems, systems of economic competition, positional games) that can be represented as twin sets of movable units (representing two or more opposing sides) and their locations can be considered as Complex Systems.

With such a problem statement for the search of the optimal sequence of transitions leading to the target state, we could use formal methods like those in the problem-solving system STRIPS (Fikes and Nilsson, 1971), nonlinear planner NOAH (Sacerdoti, 1975), or in subsequent planning systems. However, the search would have to be made in a space of a huge dimension (for nontrivial examples). Thus, in practice no solution would be obtained.

We devote ourselves to finding an approximate solution of a reformulated problem.

To create a hierarchy of dynamic subsystems, we have to use geometrical properties of the Complex System, the distance measurement (Stilman, 1993a). We have to generate a representation the hierarchy of subsystems as a Hierarchy of Formal Languages. These are Languages of Trajectories and Networks (Zones). The details of this Hierarchy are considered in other papers in these Proceedings (see, e.g., Multiagent Air Combat with Concurrent Motions). Also, a comprehensive description of these languages and their generation is presented in (Stilman, 1993a, 1993b, 1993c, 1994b, 1994d). Besides Zones considered in these papers we introduce two new types of Zones, *retreat* and *block-off* Zones. They include a target (or blocking element) with all possible trajectories of the length 1 with the beginning at the location of this element.

Network languages allow us to describe the "statics", i.e., the states of the System. In order to describe the "dynamics" of the System, i.e., the motions from one state to another, we have to regenerate the entire hierarchy of languages. Of course, it is an inefficient procedure. To improve the efficiency of applications in the search process it is important to describe the change of the hierarchy of languages (Stilman, 1994a). A study of this change helped us in modifying the hierarchy instead of regenerating it in each state. This change is represented as a mapping (translation) to some other hierarchy (actually, to the new state of the same hierarchy). Thus, the functioning of the system, in a search process, generates a tree of translations of the hierarchy of languages. This tree is represented as a string of the highest level formal language, the Language of Translations (Stilman, 1994b, 1994c).

4. Complex System of Robotic Vehicles

The robotic model can be represented as a Complex System naturally (Fig. 1). The set X represents the operational district, which could be the area of combat operation, broken into smaller square or cubic areas, “points”, e.g., in the form of the big square or cubic grid. It could be a space operation, where X represents the set of different orbits, or an air force battlefield, etc. P is the set of robots or autonomous vehicles. It is broken into two subsets P_1 and P_2 with opposing interests; $R_p(x,y)$ represent moving capabilities of different robots for different problem domains: robot p can move from point x to point y if $R_p(x, y)$ holds. Some of the robots can crawl, others can jump or ride, sail and fly, or even move from one orbit to another. Some of them move fast and can reach point y (from x) in “one step”, i.e., $R_p(x, y)$ holds, others can do that in *k* steps only, and many of them can not reach this point at all. ON(p)=x, if robot p is at the point x; v(p) is the value of robot p. This value might be determined by the technical parameters of the robot. It might include the immediate value of this robot for the given combat operation; S_i is an arbitrary initial state of operation for analysis, or the starting state; S_t is the set of target states. These might be the states where robots of each side reached specified points. On the other hand, S_t can specify states where opposing robots of the highest value are destroyed. The set of WFF {ON(p_j) = x_k} corresponds to the list of robots with their coordinates in each state. TRANSITION(p, x, y) represents the move of the robot p from the location x to location y; if a robot of the opposing side stands on y, a removal occurs, i.e., robot on y is destroyed and removed.

Aerospace robotic vehicles with different moving capabilities are shown in Fig. 1. The operational district X is the square grid of 8 × 8. The total number of squares n = 62; squares 37 and 56, representing restricted areas, e.g., neutral countries, are excluded. Robot W-CENTER (White Command & Control Flying Center) located at 88 (x = 8, y = 8), can move to any next location, e.g., from its' current location — to 78, 77, 87. Robot B-CENTER (Black Command & Control Flying Center) located at 65, can move to any next square similarly to the robot W-CENTER. Two other vehicles W-CARRIERs (White Aircraft Carriers) from 75 and 85, respectively, can move only straight ahead towards the strategic goal areas 78 and 88, one square at a time, e.g., from 75 to 76, from 76 to 77, etc. Basically, any of the squares with the coordinate y = 8 is desirable for these CARRIERs. Each of the CARRIERs carries on the top an advanced W-AS-FIGHTER (White Aerospace Fighter) which can take off only from the Aerospace Bases located in the strategic district with y = 8. After take off W-AS-FIGHTER can move in any direction, diagonally or straight forward or backward, with several squares at a time. The B-CARRIER at 35 is analogous to W-CARRIERs. It can move only straight ahead towards the strategic goal area 31 where the B-AS-FIGHTER, the cargo, can take off. The vehicle W-FIGHTER (White Aircraft Fighter) located at 53 can move only straight ahead one square at a time. The rest of Black vehicles are B-INTERCEPTOR (Black Jet-Interceptor) and B-SCOUT (Black Scout-Fighter). B-INTERCEPTOR located at 32 can move diagonally with several squares at a time, e.g., from 32 to 14 or to 54. Finally, B-SCOUT looking for a strategic information can leap forward, backward or right or left two squares at a time, e.g., from 51 it can move to 72, 63, 43.

Theoretically, B-SCOUT at 51 can reach any of the points z Œ{72, 63, 43, 32} in one step, i.e., $R_{B\text{-}SCOUT}(51,z)$ holds, while B-INTERCEPTOR can reach z {21, 43, 54, 65, 76, 87, 41, 23, 14} in one step, i.e. $R_{B\text{-}INTERCEPTOR}(32, z)$ holds. Assume that the grid is so fine that none of the vehicles can move through the square district where another vehicle is currently located. This means that in the current state B-INTERCEPTOR actually can move only to 21, 43, 54, 41, 23, 14, while B-SCOUT can leap to 72, 63, and 43.

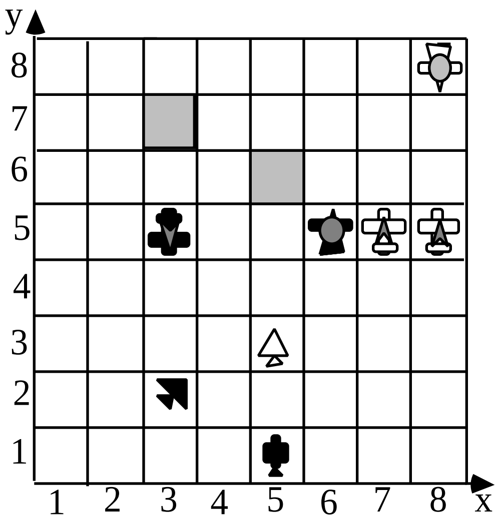

Fig. 1. 2D optimization problem for autonomous aerospace vehicles.

Assume also that robots W-CENTER, W-FIGHTER, and W-CARRIERs including, of course, their cargo, W-AS-FIGHTERs, belong to one side, while B-CENTER, B-INTERCEPTOR, B-SCOUT, and B-CARRIER with its cargo belong to the opposing side. This means that these agents belong to the sets P_1 and P_2, respectively. Each of the vehicles has powerful weapons able to destroy opposing vehicles ahead of the course, and this way move through the area where this vehicle is currently located. For example, B-CENTER from 65 can move to 85 through 75 in two steps destroying both W-CARRIERs along the way. The only difference is with the White and Black CARRIERs and W-FIGHTER. While routinely they can move only straight ahead (and be blocked by any of the friendly or opposing vehicles), they can destroy opposing vehicles at the next diagonal locations ahead of the course and then move to their respective squares. For example, W-CARRIER from 75 can destroy opposing B-CENTER at 66 and 86 and move to its' location. Obviously, each of the opposing sides must avoid loosing a respective W(B)-CENTER which means a complete destruction of the command and control battlefield communications and immediately ends the combat in a loss to this side. On the other hand, launching a totally powerful Aerospace Fighter (AS-FIGHTER) and preventing lunch of the opposing AS-FIGHTER (or destroying it) is considered as a win. The conditions considered above give us S_t, the description of target states of the Complex System. The description of the initial state S_i is obvious and follows from Fig. 1.

Assume that our time scale discretization is such that motions of the opposing sides alternate and due to the shortage of resources (which is typical in a real combat operation) or some other reasons, each side can not participate in all the motions simultaneously. It means that during the current time interval, in case the White turn, only one of the White vehicles can move. An analogous condition holds for Black. Of course, it does not mean that if one side began participating in one of the missions, it must complete it. Any time on its turn each side can switch from one mission to another, e.g., transferring resources (fuel, weapons, human resources, etc.), and later switch back.

Similar to the real world operation it is hard to predict the result of this simplified combat. However, it seems that the locations of the W-CARRIERs are advantageous in comparison with the Black agents, B-CENTER, B-INTERCEPTOR, and B-SCOUT, while B-CARRIER is too far from the strategic Aerospace Base at 31. It is likely that Black can not prevent lunches of W-AS-FIGHTERs (or destroy both of them). Is there a strategy for the Black side to win or, at least, end this combat in a draw lunching its B-AS-FIGHTER on time?

Of course, this question can be answered by a direct search employing, for example, the minimax algorithm with alpha-beta cut-offs. Theoretical estimates and experiments with computer chess programs for the similar 2D problem (in chess terms – the G. Nadareishvili endgame) showed that finding a solution of this problem requires generation of the search tree that includes about 15^{25} moves (transitions). Of course, this is beyond reasonable time constraints of the most advanced modern computers. It is very interesting to observe the dramatic reduction of search employing the Linguistic Geometry tools.

In order to demonstrate generation of the Hierarchy of Languages for this problem, we have to generate the Language of Trajectories and the Language of Zones in each state of the search. The details of generation of trajectories and Zones are considered in (Stilman, 1993b,1993c,1993d).

5. Search Generation for Air Combat

Consider how the hierarchy of languages works for the optimal control of the Air Combat System introduced above (Fig. 1). We generate the string of the Language of Translations (Stilman, 1994a) representing it as a conventional search tree (Fig.1) and comment on its generation.

In fact, this tree is close to the search tree of the G. Nadareishvili endgame generated by program PIONEER in 1977 and presented at the World Computer Chess Championship (joint event with IFIP Congress 77, Toronto, Canada). Later it was published in different journals and books, in particular in (Botvinnik, 1984).

In our comments on this generation we will emphasize the major steps and skip some technical details considered, e.g., in (Stilman, 1994c).

First, the Language of Zones in the start state is generated. The targets for attack are determined within the limit of four steps. It means that horizon H of the language $L_Z(S)$ is equal to 4, i.e., the length of the main trajectories of all Zones must not exceed 4 steps. The reasons and the algorithm for choosing the right value of the horizon are considered in (Stilman, 1994c). One of the Zones for W-CARRIER at 75, Z_{WC} is shown in Fig.1. In formal notation this Zone is as follows:

Z_{WC}=***t***(W-CARRIER, ***a***(75)***a***(76)***a***(77)***a***(78), 4)
t(B-CENTER, ***a***(65)***a***(76), 2)
t(B-CENTER, ***a***(65)***a***(66)***a***(77), 3)
t(B-CENTER, ***a***(65)***a***(66)***a***(67)***a***(78), 4)
t(B-INTERCEPTOR,***a***(32)***a***(87)***a***(78), 4)
t(W-CARRIER, ***a***(85)***a***(75), 1)
t(W-CENTER, ***a***(88)***a***(78), 1)
t(W-CENTER, ***a***(88)***a***(77), 1)
t(W-CENTER, ***a***(88)***a***(87), 1)

Search tree generation (Fig. 2) begins with the move 1. 75-76 in the most traversable “white” Zone with the vulnerable target of the highest value. This Zone Z_{WC} of W-CARRIER is shown in Fig. 3.

The order of consideration of Zones and particular trajectories is determined by the grammar of translations. The computation of move-ordering constraints is the most sophisticated procedure in the Grammar of Translations. It takes into account different parameters of Zones, trajectories, and the so-called chains of trajectories. We should keep in mind that after each move the model moves to the new current state S_c, so the entire Language of Zones, $L_Z{}^H(S_c)$, must be regenerated. With respect to efficiency of the model it is very important to solve a technical problem relative to the well known Frame Problem (McCarthy and Hayes, 1969; Fikes and Nilsson, 1971; McCarthy, 1980; Nilsson, 1980). We have to avoid recomputation of the entire language recomputing only the changing part. An approach to the formal solution of this problem is considered in (Stilman, 1994a).

The next move, 1. ... 65-66, is in the same Zone along the first negation trajectory. B-CENTER is trying to intercept motion of the W-CARRIER at 77 or 78. The interception continues: 2. 76-77 66-67 3. 77-78. Interception failed and here the grammar terminates this branch with the value of 1 (as a win of the White side). This value is given by the special state evaluation procedure built into the grammar. This procedure evaluated this state as a winning state for the White after analysis of the “traversability” of all the Zones active in this state. In particular, it figured out that the exchange at 78: 3. ... 67:78 4. 88:78 would destroy B-CENTER and, thus, it is unacceptable for Black. (Here and in the search tree symbol “:” means the removal of an element.) Moreover, the safe arrival of W-CARRIER at the strategic area 78 would cause the lunch of W-AS-FIGHTER ending the combat in a win for the White side. Also, the analysis of the Black Zones showed that Black have nothing to oppose.

The grammar initiates the backtracking climb. After the climb up to the move 2. ... 66-67 different intercepting trajectory in the same Zone (Fig. 3) has been activated ***a***(32)***a***(87)***a***(78): 2. ... 32-87. After the arrival at 87 B-FIGHTER has been destroyed by W-CENTER, and the following interception failed: 3. 88:87 66-67 4. 77-78.

The backtracking climb up to the move 3. 88:87 is interrupted at the State 2 shown in Fig. 4. This is the state where the new attacking Zone of B-SCOUT from 51 to 87 has been registered when we visited this state earlier during descent. This information has been stored to be brought to the upper levels of the search tree; the grammar stores these newly generated Zones as idle for possible activation in different states. Each backtracking move is followed by the inspection procedure, the analysis of the subtree generated in the process of the earlier search. After the climb up to the State 2 (Fig. 4), the tree to be analyzed consists of the only branch: 3. ... 66-67 4. 77-78. The inspection procedure determined that the current minimax value (+1) can be "improved" by destroying the new target at 87, the W-CENTER (in favor of the Black side). This target was staying at 87 in the analyzed subtree. The improvement can be achieved by participation of W-SCOUT from 51, i.e., by inclusion of the currently idle attack Zone with main trajectory from 51 to 87 (Fig. 4).

The motion of B-SCOUT along the main trajectory ***a***(51)***a***(63)***a***(75)***a***(87) is accompanied by the motion of intercepting element, initially as W-CARRIER, then from 78 as W-AS-FIGHTER 3. ... 51-63 4. 77-78 63-75 5. 78:75 66:75. Thus, W-SCOUT is intercepted but the newly lunched W-AS-FIGHTER is destroyed also. The current state, State 3, is shown in Fig. 5. At this state the state evaluation procedure could not generate a definite value in favor either side because two attack Zones for W-CARRIER at 85 and B-CARRIER at 35 are traversable (Fig. 5). Both Zones are activated: 6. 85-86 35-34.

Now the block-off Zone of W-CENTER should be activated in order to free the motion of W-CARRIER through 87. The exact location for the block off, 7. 87-77, is chosen in order to keep protected the most of the squares of the main trajectory: 86, 87, and 88. The race of CARRIERs continues: 7. ... 34-33 8. 86-87 33-32 9. 87-88 32-31. Both White and Black AS-FIGHTERs are ready be lunched, and the state evaluation procedure still can not terminate the branch. The current state, State 4, is shown in Fig. 6.

Among different attack Zones for W-AS-FIGHTER the Zone with the main trajectory ***a***(88)***a***(86)***a***(31) is chosen. This is a traversable "time gaining" trajectory attacking two targets simultaneously, B-CENTER at 75 and B-AS-FIGHTER at 31. After 10. 88-86 the retreat Zone of W-CENTER at 75 is activated. With two possible safe squares for retreat, 74 and 75, the wrong one is chosen first: 10. ... 75-65. New attack Zone of W-FIGHTER ***a***(53)***a***(54)***a***(65) is activated immediately because it is the time-gaining block-off trajectory as well: 11. 53-54. This motion of W-FIGHTER actually gained time. W-CENTER has been engaged and it must respond either destroying W-FIGHTER or retreating, and, thus, losing a time interval and passing a move turn to the White. W-AS-FIGHTER immediately attacks B-AS-FIGHTER along the trajectory just being blocked off: 11. ... 65:54 12. 86:31. The state evaluation procedure terminates the branch and evaluates as +1 in favor of White. The following backtracking climb up to the move 10. 88-66 where the retreat Zone of B-CENTER is activated again. Now the right square of retreat is chosen 10. ... 75-74. In absence of the vulnerable or time-gaining threats from either side the branch is terminated in a draw (0). The guilty part for this draw value is W-FIGHTER at 53. The block-off Zone registered in this terminal state as idle is stored to be activated at the upper levels of the search tree.

It seems that our preliminary estimate about easy win of the White side was incorrect. With the precise planning Black forced a draw in the variations analyzed so far. Let us continue the tree generation. The grammar initiates the backtracking climb up to the State 3 (Fig. 7). Now when we propagate the draw value as an optimum White is changing moves looking for a win. An attempt of the earlier activation of the W-CARRIER block-off Zone fails because White lose the last W-CARRIER with its valuable cargo: 6. 87-77 75:85. The optimum value is still a draw. The climb continues and move 5. 78:75 with B-SCOUT removal while W-CENTER is under direct threat is changed for W-CENTER retreat 5. 87-86. The current State 5 is shown in Fig. 7. A new Zone of B-SCOUT with the main trajectory ***a***(75)***a***(67)***a***(86) is immediately activated (Fig. 7): 5. ... 75-67 6. 78-67 66:67. B-SCOUT at 67 is intercepted by W-AS-FIGHTER while W-AS-FIGHTER itself is destroyed by B-CENTER. The state evaluation procedure does not generate a definite value in favor of either side and branch generation continues. THe following branch is quite similar to the previous long branch which includes the race of W-CARRIER from 85 and B-CARRIER from 35. The difference is that in this variation W-CENTER blocks off the main trajectory from 86 to 75: 7. 86-75, and stays there while B-CENTER is at 67 all the time (compare with Fig. 6). These new locations of White and Black CENTERs result in a draw after the arrival of both CARRIERs at the respective strategic locations, 88 and 31. The state evaluation procedure does not register vulnerable time-gaining threats and terminates this branch.

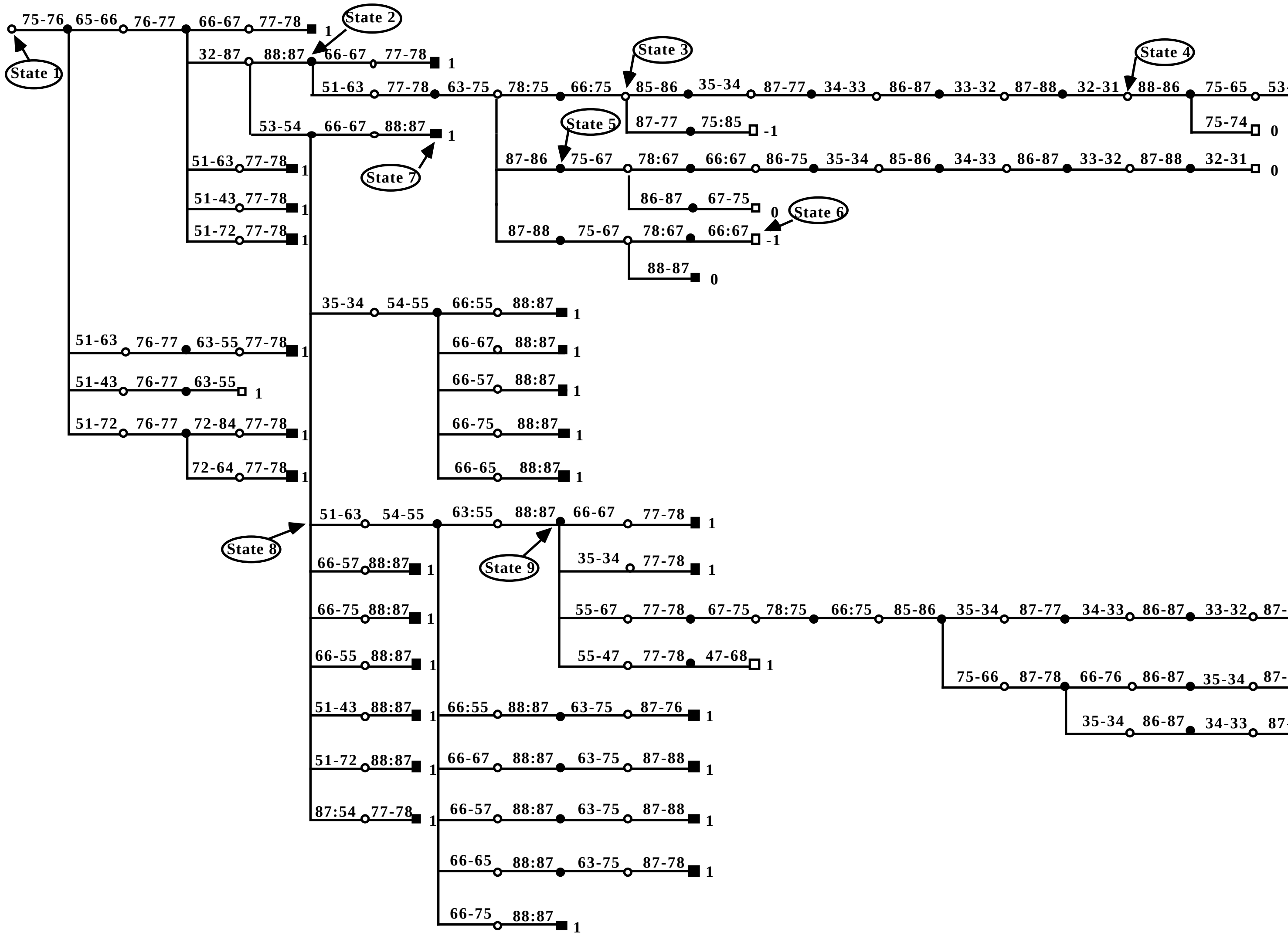

Fig. 2. Search tree for the optimization problem for aerospace autonomous vehicles within the horizon 4.

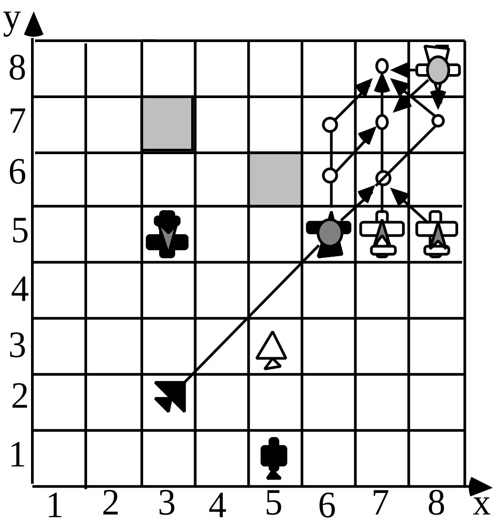

Fig. 3. State 1

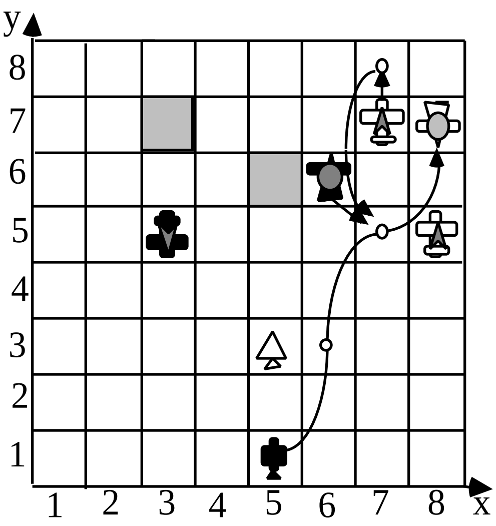

Fig. 4. State 2

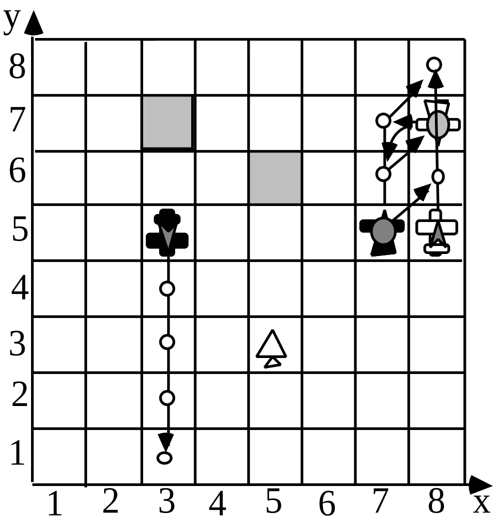

Fig. 5. State 3

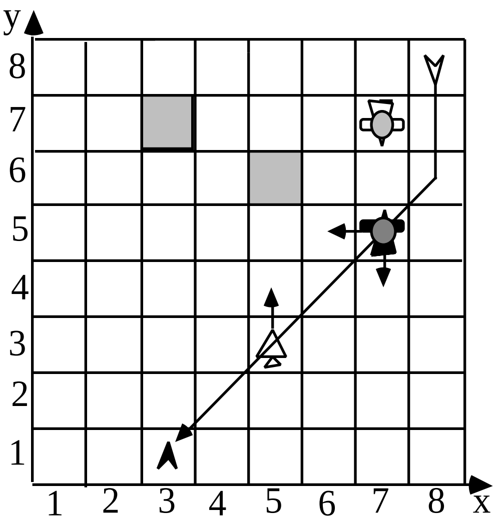

Fig. 6. State 4

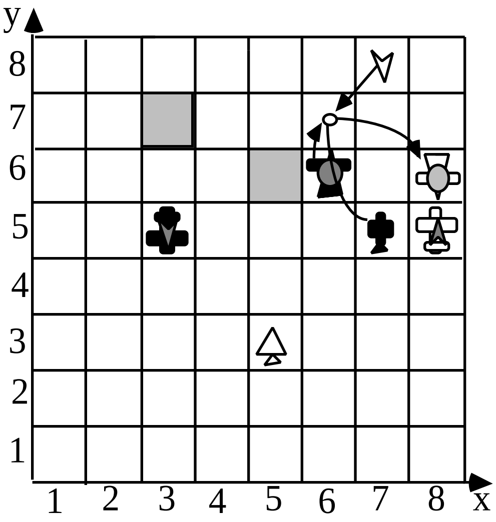

Fig. 7. State 5

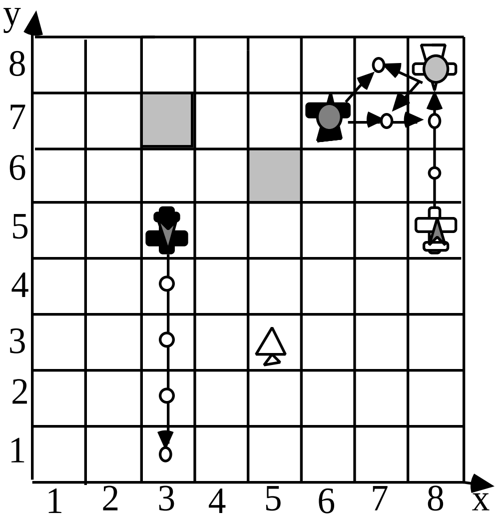

Fig. 8. State 6

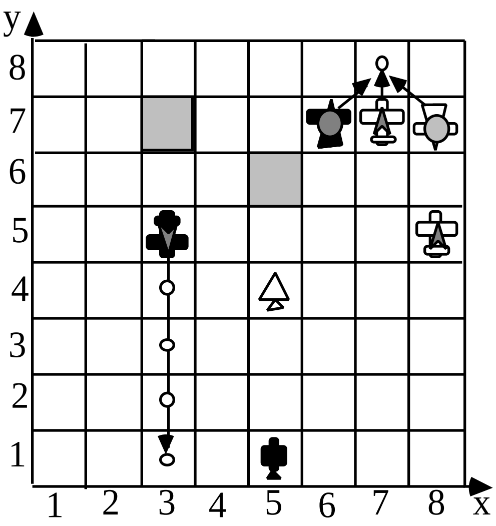

Fig. 9. State 7

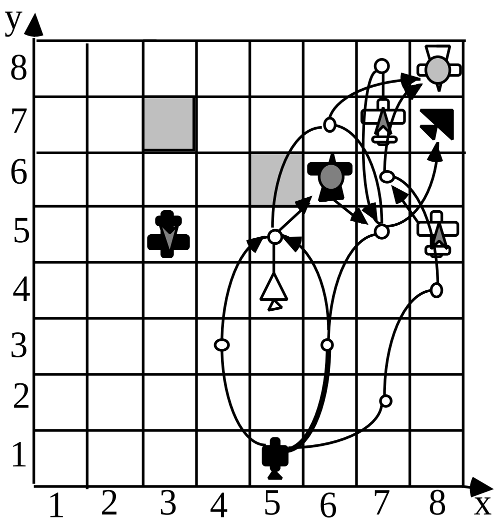

Fig. 10. State 8

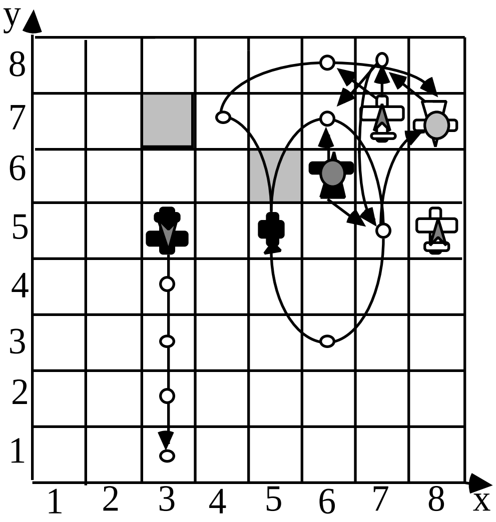

Fig. 11. State 9

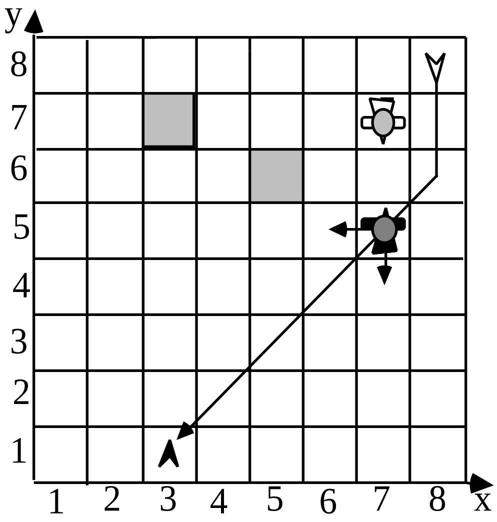

Fig. 12. State 10

The grammar initiates the backtracking climb up to the move 5. ... 75-67. At this state the tree inspection procedure activates the W-CENTER retreat Zone from 86 changing B-SCOUT interception 6. 78:67 for the only W-CENTER retreat 6. ... 86-87. THe new attack Zone of B-SCOUT with the main trajectory ***a***(67)***a***(75)***a***(87) is activated: 7. 67-75. Here the state evaluation procedure registered state repetition in the current branch (compare with the state before State 5 shown in Fig. 7), and terminated the branch with the draw value (0).

The following climb is interrupted at the state after 5. ... 63-75, and W-CENTER retreat move 6. 87-86 is changed for the last possible retreat: 6. 87-88. The new B-SCOUT attack Zone is immediately activated via ***a***(75)***a***(67)***a***(88). The intercepting trajectories are similar to the Zone shown in Fig. 7. The following variation 6. ... 75-67 7. 78:67 66:67 is terminated in the state, State 6, shown in Fig. 8. The state evaluation procedure detected that the Zone for W-CARRIER at 85 is non-traversable (because the block-off of W-CENTER is impossible) while B-CARRIER Zone from 35 is traversable, and evaluated this state as (-1) in favor of Black. The following climb and change of 7. 78:67 for W-CENTER retreat 7. 88-87 results in the state which has already occurred in the search tree and was evaluated as a draw (0).

The backtracking climb continues propagating the value of 0 (a draw) as a minimax value of the currently generated subtree. The climb stops at the move 3. 88:87, which is changed for 3. 53-54. The tree inspection procedure has chosen this move as a move of a very high preference. This is the first time when new Zone of W-FIGHTER at 53 with the main trajectory ***a***(53)***a***(54)***a***(55)***a***(66) is activated. In the backtracking climb process B-CENTER returned to 66, and now White could attack this target within the horizon 4. (The actual length of the main trajectory is 3 steps.) Moreover, this is a time-gaining motion because this is the motion in the block-off Zone of W-FIGHTER. This Zone registered in the bottom of the search tree (Fig. 6), has been idle for a long time, and now is activated as well.

The following motion continues in the Zone of W-CARRIER with participation of the intercepting and protecting elements, B-CENTER and W-CENTER: 3. ... 66-67 4. 88:87. This state is shown in Fig. 9 (State 7). It is evaluated in favor of White (+1), and the branch is terminated. From now on the current minimax value of the subtree generated so far is a win for White. Now Black try to branch. After the climb Black side activates the attack Zone of B-CARRIER at 35, while W-FIGHTER continues attack of B-CENTER: 3. ... 66-67 4. 54-55. In response, Black explore the destruction of the attacker and all possible retreats. In all these cases White continue 5. 88:87 and these branches terminated with the value in favor of White.

After multiple descents and ascents the grammar returns to the State 8 shown in Fig. 10. The tree inspection procedure activates motion of B-SCOUT along the intercepting trajectory ***a***(51)***a***(63)***a***(55) (Fig. 10). This trajectory is high preference because it partly coincides with the main trajectories of two different Zones: ***a***(51)***a***(63)***a***(75)***a***(67)***a***(88) or ***a***(51)***a***(63)***a***(75) ***a***(67)***a***(88) with W-CENTER as a target. Moreover, this motion is also the motion along the main trajectory in the control Zone ***a***(51)***a***(63)***a***(75)***a***(87) with the square 87 as a location of the future target, W-CENTER, whose arrival is expected by the tree inspection procedure. As usual, this control Zone was registered in the bottom of the search tree and kept idle until now. Thus 3. ... 51-63 should be considered as a highly time-gaining move. The State 9 generated after 3. ... 51-63 4. 54-55 63:55 5. 88:87 is shown in Fig. 11.

After the futile attempts to continue interception of W-CARRIER by W-CENTER or attack by B-CARRIER, the grammar returns to the State 9. At this moment the tree inspection procedure activates new attack Zone of B-SCOUT from 55 to 87. Among the bundle of such Zones (Fig. 11) the Zone with the most traversable main trajectory ***a***(55)***a***(67)***a***(75)***a***(87) is picked up. After 5. ... 55-67 6. 77-78 67-75 7. 78:75 66:75, the state is exactly the same as State 3 (Fig. 5) generated earlier in the search tree. The only difference is that in the current state there is no W-FIGHTER at 53. As we know the minimax value for the State 3 propagated from the bottom of the search subtree was a draw (0). So, it seems that Black which is currently looking for this value have found one. This, probably, means that after 3. 53-54, Black eventually have found the right variation leading to a draw. But, because of the different location of W-FIGHTER mentioned above we can not just consider this state as the state visited before, terminate this branch, and assign the value. Analogously to the State 3 (on descent), the state evaluation procedure can not give a definite value to this state, so the branch continues. All the following moves, the CARRIERs race, are exactly the same as in the earlier branch generated from the State 3. The race is complete when both CARRIERs have reached their respective Aerospace Bases. The corresponding State 10 is shown in Fig. 12. The only difference of this state with the State 4 (Fig. 6) is the absence of W-FIGHTER at 53. But this tiny change makes big difference. The motion of W-AS-FIGHTER along the time-gaining trajectory ***a***(88)***a***(86)***a***(31) is a simultaneous attack both B-CENTER and B-AS-FIGHTER. This means that at least one of the targets will be destroyed. The continuation is as follows: 12. 88-86 75-65 (or 12. ... 75-74) 13. 86:31. In both variations W-AS-FIGHTER is destroyed and they are terminated with the value (+1) in favor of White. Thus, despite of this long 25-move(!) resistance, Black achieved nothing. The current minimax value is still in favor of White.

The following climb and branching when Black tries, e.g., most efficiently activate the retreat Zone of B-CENTER from 75 at the upper levels of the search tree or explore different B-SCOUT attack trajectories from 51, does not change the minimax value. The following tree generation does not even yield a “better” (longer) resistance variation than the best variation generated so far. Basically, this is the optimal variation which is likely to be followed by both sides in the actual battlefield. In order to generate this branch the grammar used the information, the key networks (W-FIGHTER retreat Zone) learned at the bottom of the search tree in the previously generated non-optimal branches.

The search tree generated by the grammar consists of 152 moves. Obviously, this is a dramatic reduction in comparison with billion-move trees generated by conventional search procedures and still insufficient for solving this problem.

6. Discussion

The example considered in this paper demonstrates the power of the Linguistic Geometry tools that allowed to transfer heuristics discovered in the 2D problem domain of positional games, to another domain of simplified aerospace robotic vehicles. The conventional approaches employing search algorithms with alpha-beta pruning require approximately 15^{25} move search tree to solve this problem, while the tree presented in this paper consists of about 150 moves. Moreover, the branching factor of this search, i.e., the average number of moves in each node, is about 1.12(!) while the depth of the search required to solve this problem must be at least 25 moves. This means that the algorithm is actually goal-oriented, i.e., it approaches the goal almost without branching to different directions. Looking at the complexity of the hierarchy of languages which represents each state in the search process, it is very likely that the growth from the problems with the lesser number of agents with limited moving capabilities (Stilman, 1994b, 1994c) to the current essentially more complex problem is linear with the factor close to one. This means that the complexity of the entire algorithm may be about linear with respect to the length of the input.

At the same time the simplified aerospace navigation problem considered here is still very close to the original chess domain. It is possible to predict that the power of Linguistic Geometry goes far beyond these limits. The definition of the Complex System (Section 3.1) is generic enough to cover a variety of different problem domains. The core component of this definition is the triple X, P, and R_p. Thus, looking at the new problem domain we have to define X, the finite set of points – locations of elements. We do not impose any constraints to this set while the operational district X considered in this paper as well as the original chess board have different extra features, e.g., 2D space connectivity, which is totally unimportant for these problems. Thus, for example, we can consider X as a set of orbits where the elements are in permanent motion with respect to each other. The moving capabilities of elements P in our example, i.e., the binary relations R_p, are non-sophisticated. This is exactly the place for introduction of the variable speed, the gravity impact, the engine impulse duration, etc.

Also, it should be noted that we introduced some additional constraints for the Complex System in example considered in this paper. These are requirements of the motion alternation for the opposing sides and participation of the only element in each motion. This introduction was done only for a transparent display of ideas and advantages of Linguistic Geometry. The generic definition of the Complex System (Section 3.1) does not include these constraints.

The development of Linguistic Geometry towards aerospace applications will encompass the discovery of geometrical properties of subsystems, details of interactions between the elements within subsystems and between different subsystems, the effect of this complex hierarchical structure on the search reduction.

References

Albus, J. (1991). Outline for a Theory of Intelligence. *IEEE Trans. on Systems, Man and Cybernetics* (pp. 473-509), 3.

Boddy, M. and Dean, T. (1989). Solving Time-Dependent Planning Problems, *Proc. of the 11th Int. Joint Conf. on AI*, (pp. 979-984).

Botvinnik, M.M. (1984). *Computers in Chess: Solving Inexact Search Problems.* Springer Series in Symbolic Computation, New York: Springer-Verlag.

Botvinnik, M., Petriyev, E., Reznitskiy, A., et al. (1983). Application of New Method for Solving Search Problems For Power Equipment Maintenance Scheduling. *Economics and Mathematical Methods* (pp. 1030-1041), 6, (in Russian).

Chapman, D. (1987). Planning for conjunctive goals. *Artificial Intelligence* 32(3).

Chomsky, N. (1963). Formal Properties of Grammars. in *Handbook of Mathematical Psychology*, eds. R.Luce, R.Bush, E. Galanter., vol. 2 (pp. 323-418). New York: John Wiley & Sons.

Chung, J., Liu, J. and Lin, K., (1990). Scheduling Periodic Jobs That Allow Imprecise Results. *IEEE Transactions on Computers*, (pp. 1156-1174), 39(9).

Drabble, B., (1991) Spacecraft Command and Control Using Artificial Intelligence Techniques, *J. of the British Interplanetary Society*, Vol. 44, (251-254).

Durfee, E.H. and Lesser, V.R. (1987). Incremental Planning to Control a Blackboard-Based Problem Solver. *Proceedings of the Workshop on Space Telerobotics*, Vol. 3, NASA Jet Propulsion Lab, July, (pp. 91-99).

Feder, J. (1971). Plex languages. *Information Sciences*, 3: 225–241.

Fikes, R.E. and Nilsson, N.J. (1971). STRIPS: A New Approach to the Application of Theorem Proving in Problem Solving. *Artificial Intelligence* 2: 189–208.

Fu, K.S. (1982). *Syntactic Pattern Recognition and Applications*, Prentice Hall, Englewood Cliffs.

Garey, M.R. and D.S.Johnson D.S. (1991). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Co., San Francisco.

Giarratano, J.C., (1991). CLIPS User's Guide, NASA Johnson Space Center, Information Systems Directorate (JSC-25013).

Ginsburg, S. (1966). *The Mathematical Theory of Context-Free Languages*, McGraw Hill, New York.

King, R. D. (1993). Rule Based Approach to Hierarchical Grammars for Geometrical Reasoning, Master Thesis, Dept. of Computer Science, Univ. of Colorado at Denver.

Knoblock, C.A. (1990). Learning Abstraction Hierarchies for Problem Solving, *Proc. of the 8th AAAI Conf.*, (pp.923-928), Menlo Park, CA.

Knuth, D.E. (1968). Semantics of Context-Free Languages. *Mathematical Systems Theory*, (pp. 127–146), 2.

Korf, R.E. (1990). Real-Time Heuristic Search, *Artificial Intelligence*, (pp. 189-211), 42(2-3).

Leitmann, G., (1990). *Optimization Techniques with Applications to Aerospace Systems*, Academic Press.

Lesser, V.R., Pavlin, J., and Durfee, E. (1988). Approximate Processing in Real-Time Problem Solving, *AI Magazine*, (pp. 49-62), 9(1).

Lirov Y., Rodin, E.Y., McElhaney, B.G., and Wilbur, L.W. (1988). Artificial Intelligence Modeling of Control Systems, *Simulation*, (pp. 12-24), 50(1).

Mathews, E. R. (1993). An Implementation of the Grammars of Trajectories and Zones in C Language, Master Thesis, Dept. of Mathematics, Univ. of Colorado at Denver.

McAllester, D. and Rosenblitt, D. (1991). Systematic Non-Linear Planning, *Proc. of AAAI-91*, (pp. 634-639).

McCarthy, J. (1980). Circumscription–A Form of Non-Monotonic Reasoning. *Artificial Intelligence*,(27-39), 13.

McCarthy, J. and Hayes, P.J. (1969). Some Philosophical Problems from the Standpoint of Artificial Intelligence. *Machine Intelligence* (pp. 463–502), 4.

Mesarovich, M.D. and Takahara Y. (1989). *Abstract Systems Theory*, Berlin: Springer-Verlag.

Narasimhan, R.N. (1966). Syntax–Directed Interpretation of Classes of Pictures. *Comm. of ACM* (pp. 166–173), 9.

Nilsson, N.J. (1980). *Principles of Artificial Intelligence*, Palo Alto, CA: Tioga Publ.

Pavlidis, T. (1977). *Structural Pattern Recognition*, New York: Springer-Verlag.

Pigeon, A., Howard, G., and Seaton B. (1992) Operational Aspects of Space craft Autonomy, *J. of the British Interplanetary Society*, Vol. 45, (pp. 87-92).

Reznitskiy, A.I. and Stilman, B. (1983). Use of Method PIONEER in Automating the Planning of Maintenance of Power-Generating Equipment. *Automatics and Remote Control* (pp. 147-153), 11, (in Russian).

Rodin E. (1988). Semantic Control Theory, *Applied Mathematical Letters*, (pp. 73-78), 1(1).

Rodin E., Garsia-Ortiz et al. (1993). Application of Semantic Control to a Class of Pursue-Evader Problems, *Computers and Mathematics with Applications*, 26(5).

Rosenfeld, A. (1979). *Picture Languages, Formal Models for Picture Recognition*, Academic Press.

Rozenkrantz, D.J. (1969). Programmed Grammars and Classes of Formal Languages, *J. of the ACM* (pp. 107–131), 1.

Sacerdoti, E.D. (1975). The Nonlinear Nature of Plans, *Proc. Int. Joint Conference on Artificial Intelligence*.

Shaw, A.C. (1969). A Formal Picture Description Scheme as a Basis for Picture Processing System, *Information and Control* (pp.9-52), 19.

Simon, H.A. (1980). *The Sciences of the Artificial*, 2-nd ed., The MIT Press, Cambridge, MA.

Stefik, M. (1981). Planning and meta-planning (MOLGEN: Part 2),"*Artificial Intelligence* (pp. 141-169), 2.

Stilman, B. (1977). The Computer Learns. in Levy, D., *1976 US Computer Chess Championship* (pp. 83-90). Computer Science Press, Woodland Hills, CA.

Stilman, B. (1985). Hierarchy of Formal Grammars for Solving Search Problems. In *Artificial Intelligence. Results and Prospects*, *Proceedings of the International Workshop* (pp. 63-72), Moscow, (in Russian).

Stilman, B. (1993a). A Linguistic Approach to Geometric Reasoning," *Int. J. Computers and Mathematics with Applications* (pp. 29-57), 26(7).

Stilman, B. (1993b). Network Languages for Complex Systems, *Int. J. Computers and Mathematics with Applications* (pp. 51-79), 26(8).

Stilman, B. (1993c). Syntactic Hierarchy for Robotic Systems, *Integrated Computer-Aided Engineering* (pp. 57-81), 1(1).

Stilman, B. (1993d). A Formal Language for Hierarchical Systems Control, *Languages of Design* (333-356), 1(4).

Stilman, B. (1994a). Translations of Network Languages. *Int. J. Computers and Mathematics with Applications* (pp. 65-98), 27(2).

Stilman, B. (1994b). A Formal Model for Heuristic Search. *Proc. of the 22nd Annual ACM Computer Science Conf.*, (pp. 380-389), March 8-10, Phoenix, AZ.

Stilman, B., (1994c). A Linguistic Geometry for Space Applications, *Proc. of the 1994 Goddard Conference on Space Applications of Artificial Intelligence*, (pp. 87-101), NASA Goddard Space Flight Center, Greenbelt, MD, USA, May 1994.

Strosnider, J.K., Paul, C.J. (1994). A Structured View of Real-Time Problem Solving, *AI Magazine*,(45-66), 15(2).